Car Price Prediction with Machine Learning

Created by: Jangita Takeshwar

Introduction

Predicting car prices accurately is crucial for buyers and sellers in the automobile industry. This project leverages machine learning techniques to predict the selling price of a car based on various features such as year of manufacture, present price, kilometers driven, fuel type, selling type, transmission, and owner history.

Libraries Used

The following Python libraries are used in this project:

- pandas - For data manipulation and analysis
- numpy - For numerical operations
- matplotlib.pyplot & seaborn - For data visualization
- sklearn.model_selection - For splitting the dataset into training and testing sets
- sklearn.preprocessing - For encoding categorical variables and feature scaling
- sklearn.ensemble - For implementing the Random Forest Regressor model
- sklearn.metrics - For evaluating the model performance

Code Implementation

Importing Necessary Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

Loading the Dataset

```
data = pd.read_csv('/kaggle/input/car-data/car data.csv')
print(data.head())
```

Data Preprocessing

```
data = data.dropna()

label_encoder = LabelEncoder()
data['Fuel_Type'] = label_encoder.fit_transform(data['Fuel_Type'])
data['Selling_type'] = label_encoder.fit_transform(data['Selling_type'])
data['Transmission'] = label_encoder.fit_transform(data['Transmission'])
```

Feature Selection

```
X = data[['Year', 'Present_Price', 'Driven_kms', 'Fuel_Type', 'Selling_type', 'Transmission',
'Owner']]
y = data['Selling_Price']
```

Splitting the Data

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Feature Scaling

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Model Training

```
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Model Evaluation

```
y_pred = model.predict(X_test)

mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error: {mae}")
print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

Data Visualization

```
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
sns.histplot(data['Selling_Price'], bins=20, kde=True, color='blue')
plt.title('Distribution of Selling Prices')

plt.subplot(1, 2, 2)
sns.heatmap(data.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Feature Correlation Heatmap')

plt.tight_layout()
plt.show()
```

Actual vs Predicted Prices

```
plt.figure(figsize=(8, 5))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.7, color='red')
```

```
plt.xlabel("Actual Selling Price")
plt.ylabel("Predicted Selling Price")
plt.title("Actual vs. Predicted Selling Price")
plt.show()
```

Prediction Example

```
example = np.array([[2017, 9.29, 37000, 1, 0, 1, 0]])
example_scaled = scaler.transform(example)
predicted_price = model.predict(example_scaled)
print(f"Predicted Selling Price: {predicted_price[0]}")
```

Conclusion

This project successfully predicts car prices using machine learning. The Random Forest Regressor demonstrated strong predictive power, as indicated by the model evaluation metrics. Future improvements could involve trying other models such as Gradient Boosting or incorporating additional features to enhance accuracy.